

**CyberDog**

**AppleSearchClient**  
**AppleSearchItem**  
**AppleSearchService**  
**AppleSearchViewer**  
**ArticleViewer**

**Book**

The Book class represents the user's favorite place book and will be the root part of a Cyberdog document. It is chiefly responsible for managing the pages in the book.

*fields*  
 short                                      fCurrentPage  
 PageList\*                                fThePages  
 CyberBookExtension                      \*fCyberBookExtension  
*methods*  
 void                                      AddUserPage()  
 void                                      AddCyberItem(CyberItem\* theItem)  
 void                                      Internalize()  
 void                                      Externalize()

**BookExtension**

Shared Library

This class represents the public API to the Book class (which we do not want to expose to developers). It contains any methods necessary for a part developer to access the favorite places book.

*comments*  
 OpenDoc Extension -  
*fields*  
 Book                                      \*fBook  
*methods*  
 void                                      Initialize(Book \*theBook)  
 void                                      AddCyberItem(CyberItem \*theItem)

**CoverPage****CyberBrowser**

*fields*  
 TreeView\*                                fTheHieararchy  
*methods*

**CyberBrowser****CyberItem**

Shared Library

Small, self-contained class which completely describes a location on the Internet. CyberItems make up the contents of the log and the user pages of the favorite places book. Each CyberPart displays information for a single CyberItem. CyberItem is an abstract superclass and must be subclassed for each type of item that may appear in the log or favorite places book. The TelnetItem subclass, for example, will contain additional information about the terminal settings for that session.

*fields*  
 URL                                      fURL  
 StringPtr                                fDisplayName  
 IconFamily                               fIconFamily  
 CyberType                                fType  
 WindowState                            fAlaska  
 CyberItemFamily\*                        fFamily  
*methods*  
 void                                      Internalize()  
 void                                      Externalize()  
 Boolean                                IsEqual(CyberItem\* theItem)  
 CyberItem\*                               Copy()

```

        StringPtr      GetProperty(Book* theBook, Property theProperty)
        void           MakePart()
CyberItemFamily
Shared Library

```

User pages display lists of CyberItems, but not all CyberItems are displayed the same way. Newsgroups, for example, display the number of unread articles while a Telnet session displays only the word "Telnet." Subclasses of CyberItemFamily (an abstract superclass) know how to display individual CyberItems on a user page.

```

CyberItemList
CyberPart
Shared Library

```

Abstract superclass representing a part handler for any type of data that will be displayed in Cyberdog. Each CyberPart is responsible for displaying and handling events for a single CyberItem.

```

        fields
            CyberItem*    fMe
            BookExtension  *fBookExtension
        methods
CyberService
Shared Library

```

Abstract superclass representing a service that can be placed on the cover of the Favorite Places Book

```

        fields
            Str255        fName
            short          fPictResID
        methods
            void           LaunchService()
            void           DrawOnCover()
            void           GetMenuItemName(Str255 name)

```

```

DefaultFamily
FTPBrowser
FTPBrowser
FTPItem
FTPService
GIFItem
GIFViewer
GopherBrowser
GopherBrowser
GopherDirItem
GopherService
GopherStream
HTTPStream
JPEGItem
JPEGViewer

```

**Log**

The Log class controls the behavior of the automatic log. It contains data structures for viewing the log entries (which are CyberItems) as a hierarchy or as a list, and knows how to add and display new CyberItems.

```

        fields
            Tree*          fHierarchicalView
            CyberItemList* fListView
            ViewByType     fCurrentViewByType
        methods
            void           AddCyberItem(CyberItem* theItem, CyberItem* itsParent)

```

**LogExtension**  
Shared Library

This class represents the public API to the log (which we do not want to expose to developers). It contains any methods necessary for a part developer to access the log.

*fields*

*methods*

void

AddCyberItem(CyberItem\* theItem, CyberItem\* itsParent)

**Mosaic**

**MosaicItem**

**MosaicViewer**

**MovieItem**

**MovieViewer**

**NewsgroupBrowser**

**NewsgroupFamily**

**NewsgroupItem**

**NewsgroupService**

**NewsgroupViewer**

**NotificationPage**

**Page**

Abstract superclass defining behavior common to all pages in the Favorite Places Book.

*fields*

Str255

fTitle

*methods*

void

TearOff();

**PreferencesPage**

**SoundItem**

**SoundViewer**

**Stream**

Abstract superclass that represents a data stream in which information is downloaded from a remote source a chunk at a time. Use of this stream allows viewers to display data as it is downloaded without needing to know the underlying protocol used to obtain the data.

**StreamViewer**

Abstract superclass for a Viewer that needs to download data from a stream.

**TelnetItem**

**TelnetService**

**Text (Zaky)**

**TextItem**

**TextViewer**

**Tree**

Data structure representing a generic hierarchy. This structure will be used for storing the hierarchical view of the log and for representing hierarchical data spaces such as Gopher, FTP, and the newsgroup hierarchy.

*fields*

*methods*

void

AddItem(Element\* theItem, Element\* itsParent);

void

ExpandItemAt(short index);

void

CollapseItemAt(short index);

short

NumItems();

**TreeView**

A TreeView object puts a human interface on the data stored in its fTheTree field. TreeViews know how to display hierarchical lists, respond to events, and handle clicks on turney-triangles.

*fields*

Tree\*

fTheTree

```

    methods
        void Draw();
        void DoMouseDown(EventRecord* theEvent);
        void DoKeyDown(EventRecord* theEvent);
        void DoDoubleClick(EventRecord* theEvent);
UserPage
    fields
        Str255 fTitle
        CyberItemList* fContents
    methods
        void AddCyberItem(CyberItem* theItem)

```